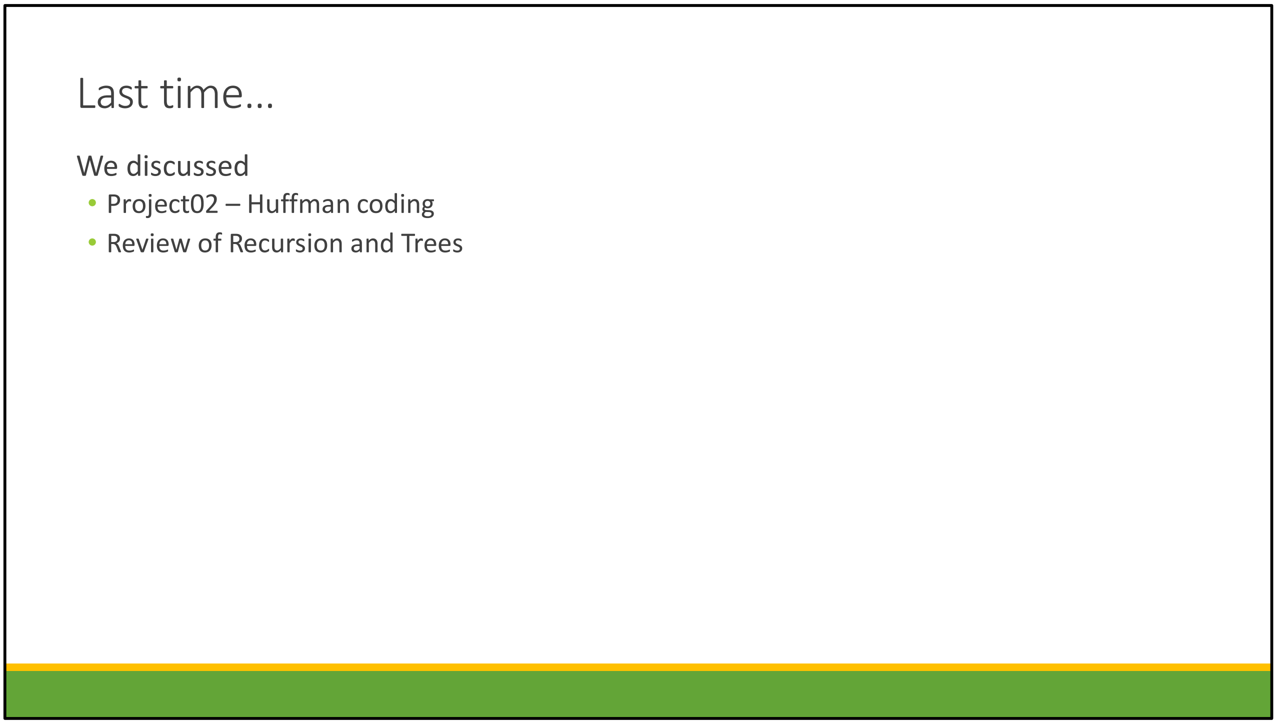


Last time…

We discussed

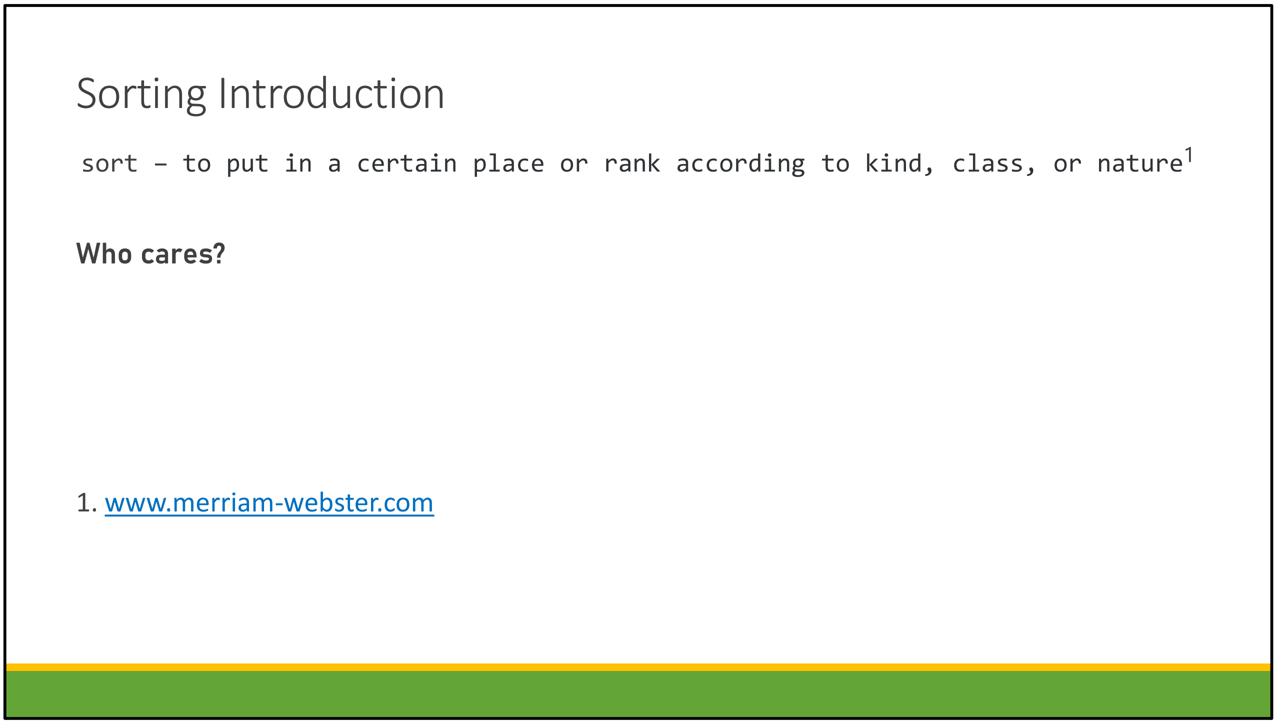
- Project02 – Huffman coding
- Review of Recursion and Trees

Sorting Algorithms

CS112 –
Java
Programming

Fall 2022

Good news no HW today on sorting—have Project02! But will be sort questions on final



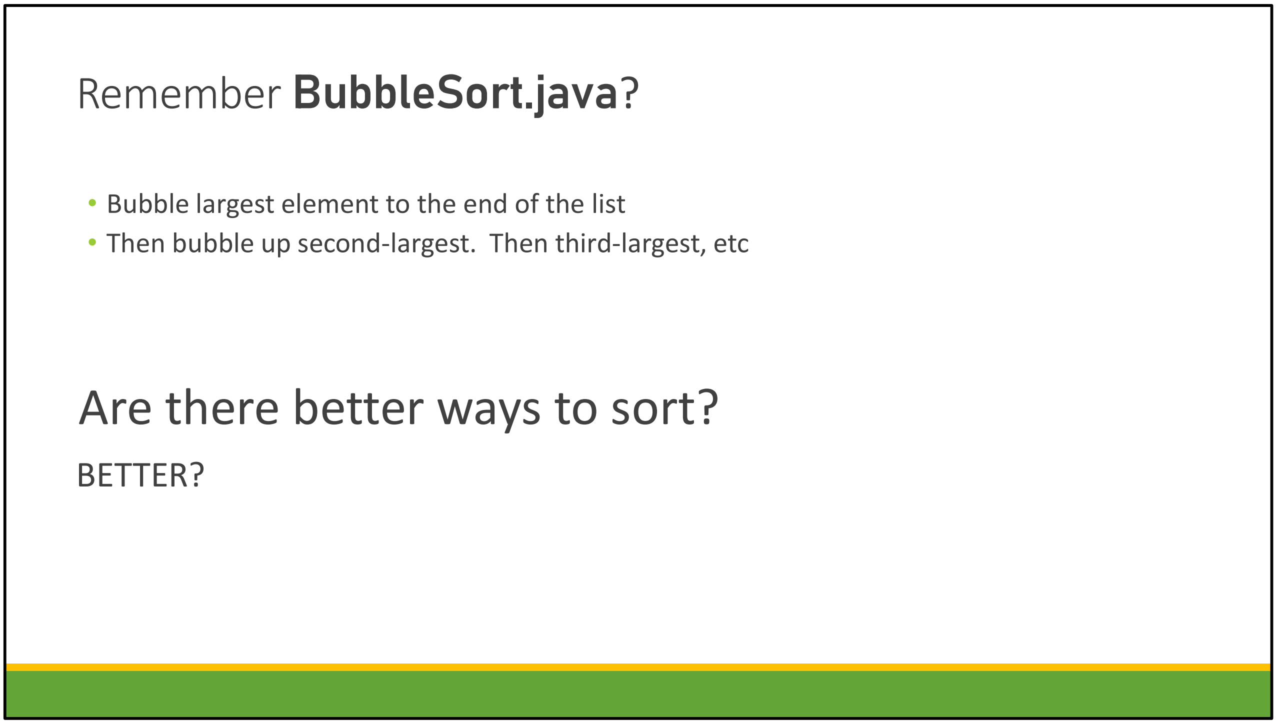
Sorting Introduction

`sort – to put in a certain place or rank according to kind, class, or nature`[1]

Who cares?

[1] www.merriam-webster.com

Pack boxes into container: "biggest rocks first".
Pack compute jobs onto server farm.
Schedule data packets onto shared channel.
Search sorted list faster than unsorted with Binary Search…



Remember **BubbleSort.java**?

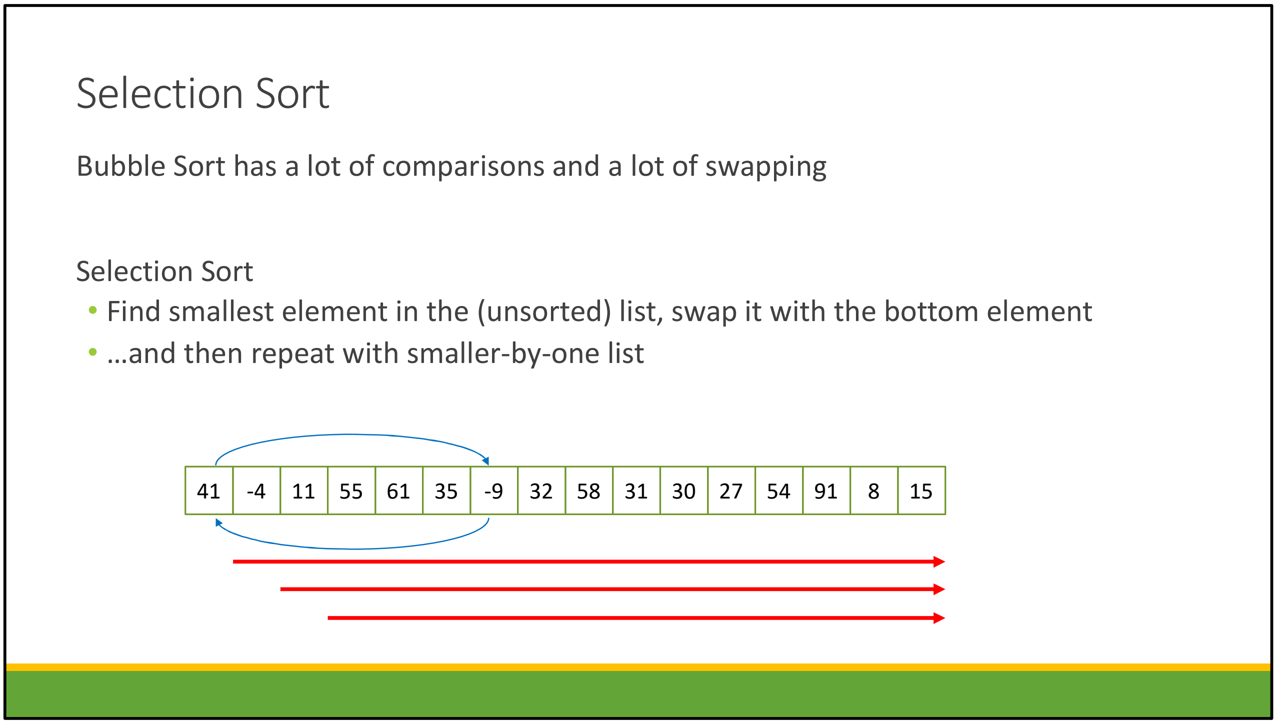
- Bubble largest element to the end of the list
- Then bubble up second-largest. Then third-largest, etc

Are there better ways to sort?

BETTER?

Write up in SCRATCH.java. Class type-along.

Better? FASTER, small memory use, simple SW to write & debug



Selection Sort

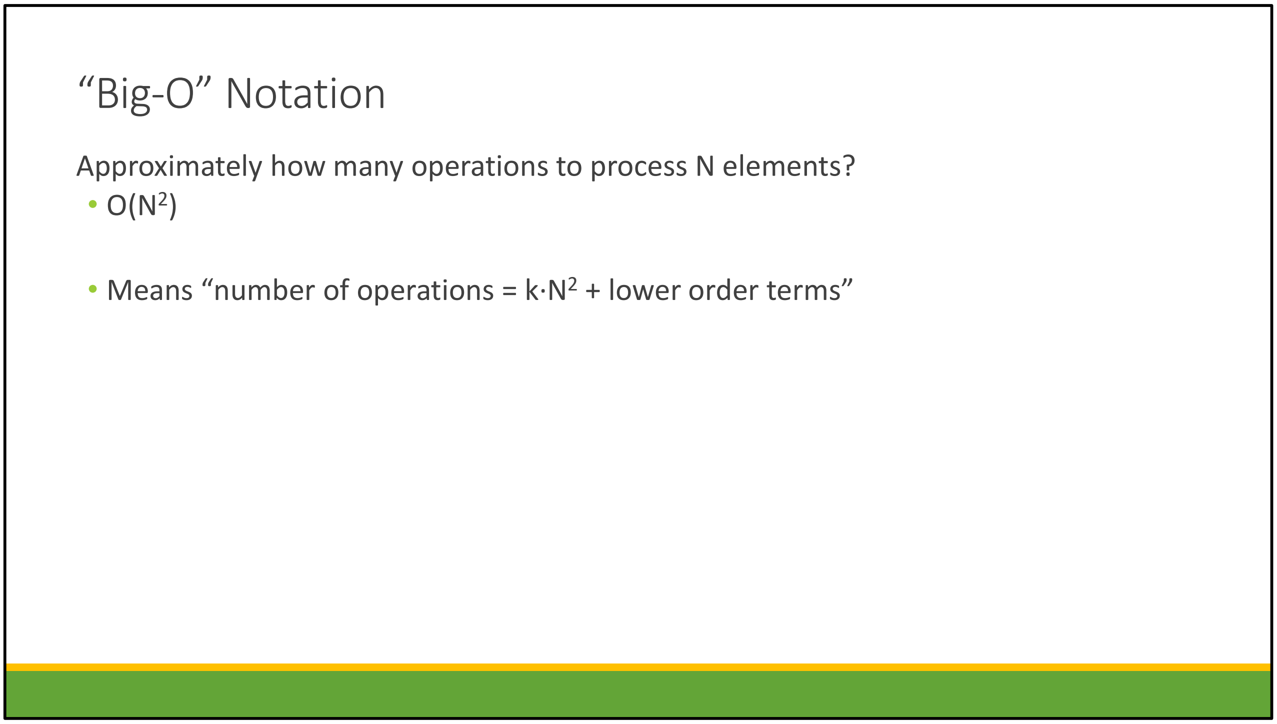
Bubble Sort has a lot of comparisons and a lot of swapping

Selection Sort

- Find smallest element in the (unsorted) list, swap it with the bottom element
- …and then repeat with smaller-by-one list

41	-4	11	55	61	35	-9	32	58	31	30	27	54	91	8	15

Write code. Then review code w COUNTED # OF COMPARES & ASSIGNS. Measure time!



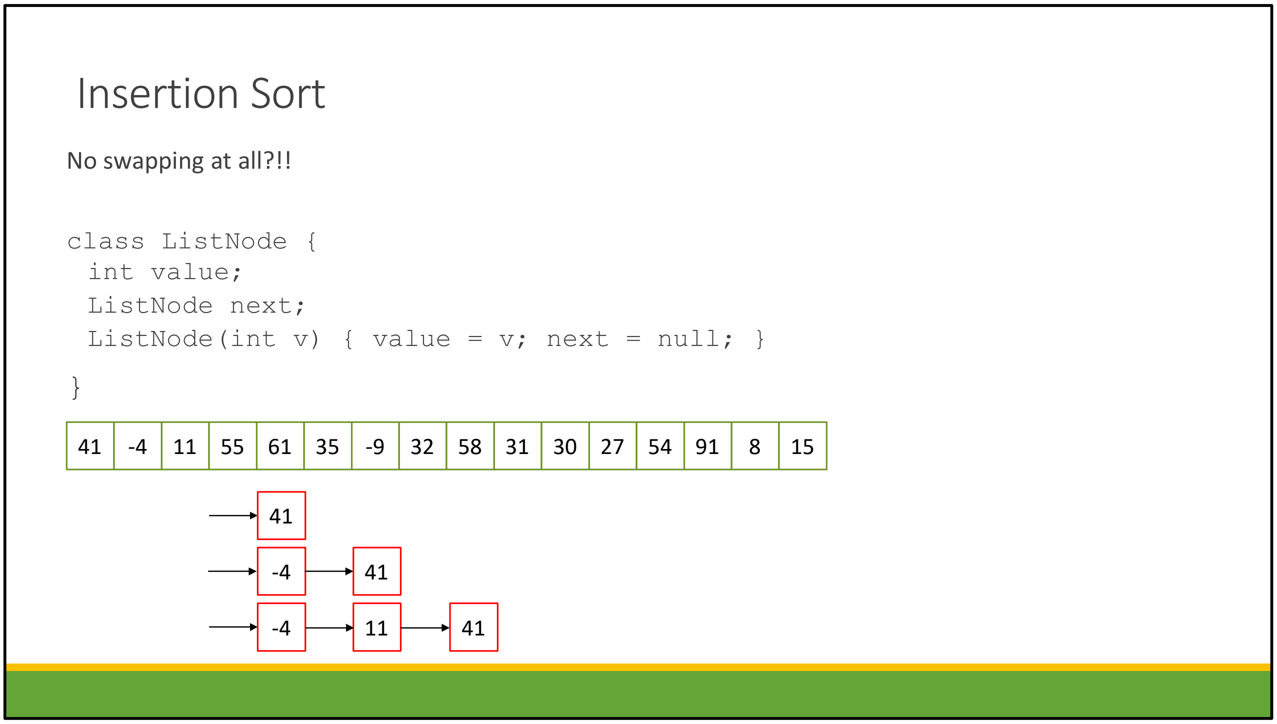
“Big-O” Notation

Approximately how many operations to process N elements?

- O(N^2)

- Means “number of operations = $k \cdot N^2$ + lower order terms”

Analyze for Bubble Sort, Selection Sort
Gives order of magnitude



Insertion Sort

No swapping at all?!!

```
class ListNode {
 int value;
 ListNode next;
 ListNode(int v) { value = v; next = null; }
}
```

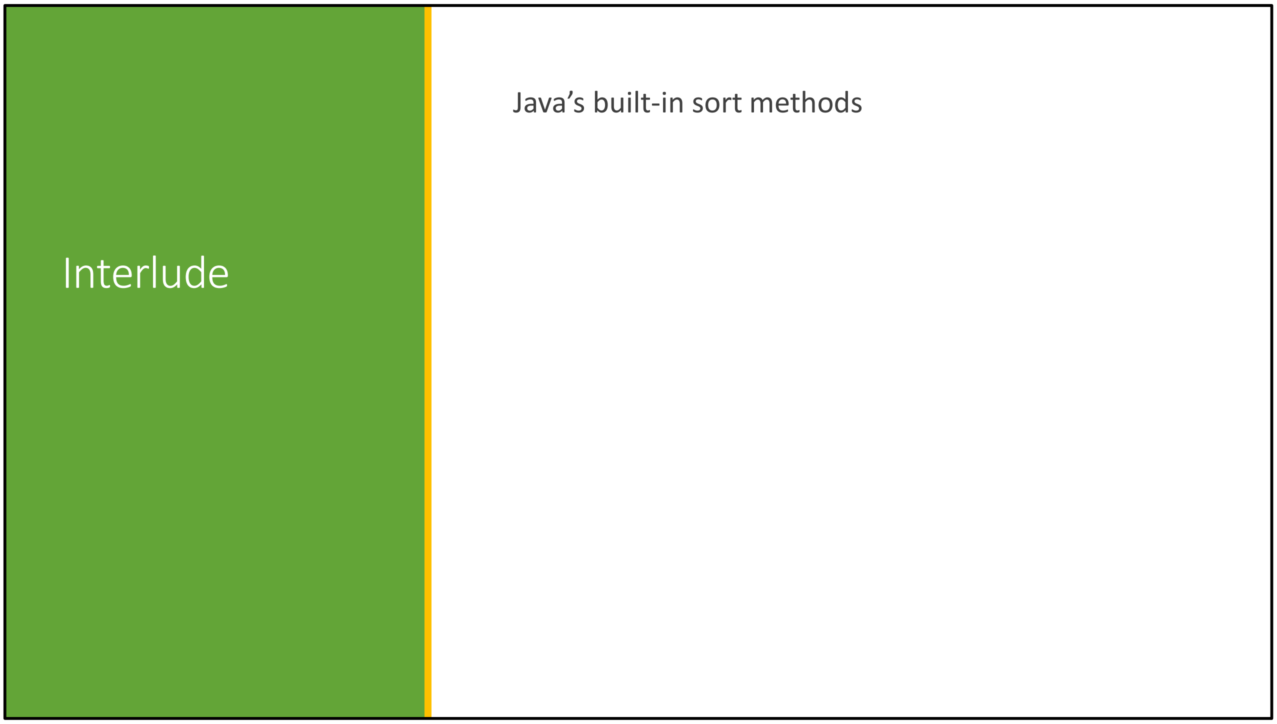
41	-4	11	55	61	35	-9	32	58	31	30	27	54	91	8	15

→ 41

→ -4 → 41

→ -4 → 11 → 41

Look @ & run SW: no swaps: why so slow? Expensive operations.
EVER USE Insertion Sort? If data "almost" sorted, use doubly linked list, insert at end, sort in O(N)!



Interlude

Java’s built-in sort methods

Look at SCRATCH: inherits Comparable, compareTo() function. Collections.sort(). That’s it!

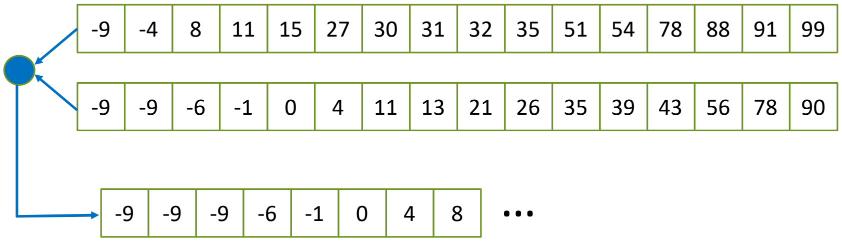
Java’s sort? Timsort

High Performance Sorting Algorithms

MergeSort

Remember lab problem from 3 days ago?

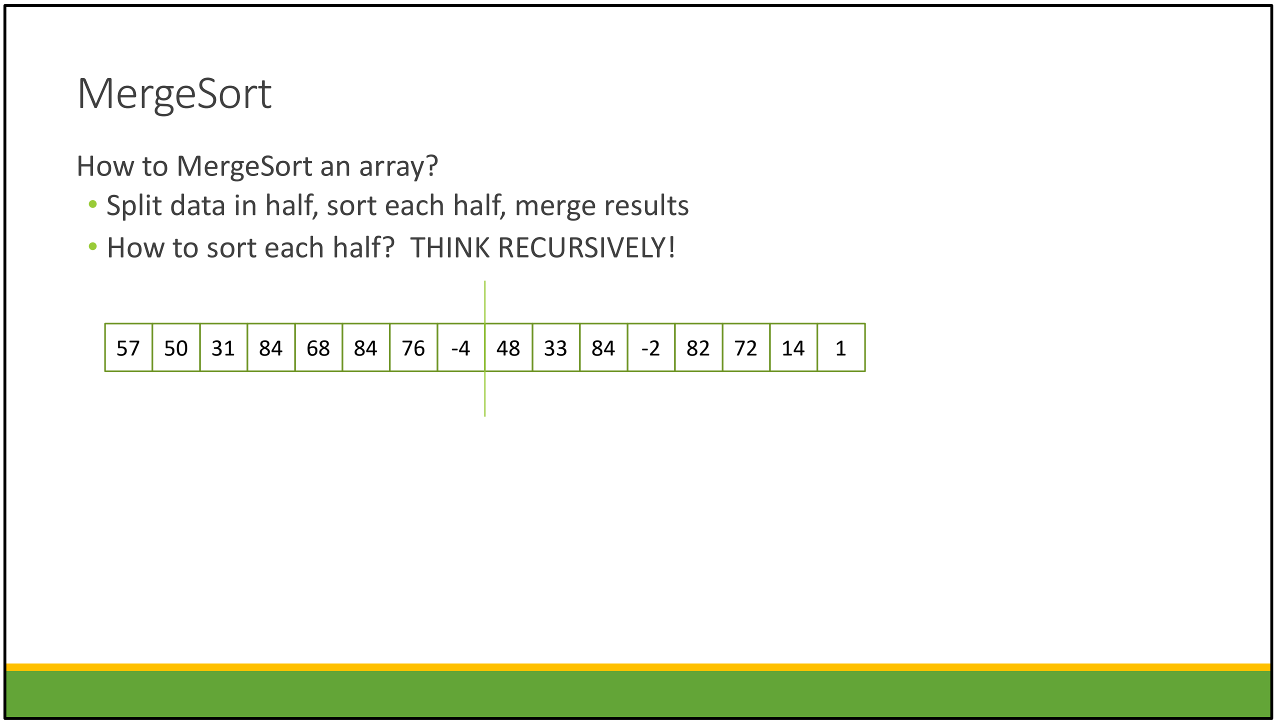
- Several sorted lists: merge into one



-9	-4	8	11	15	27	30	31	32	35	51	54	78	88	91	99
-9	-9	-6	-1	0	4	11	13	21	26	35	39	43	56	78	90

-9	-9	-9	-6	-1	0	4	8	…

ONLY PULL FROM FRONT OF LISTS. O(N), very cheap



MergeSort

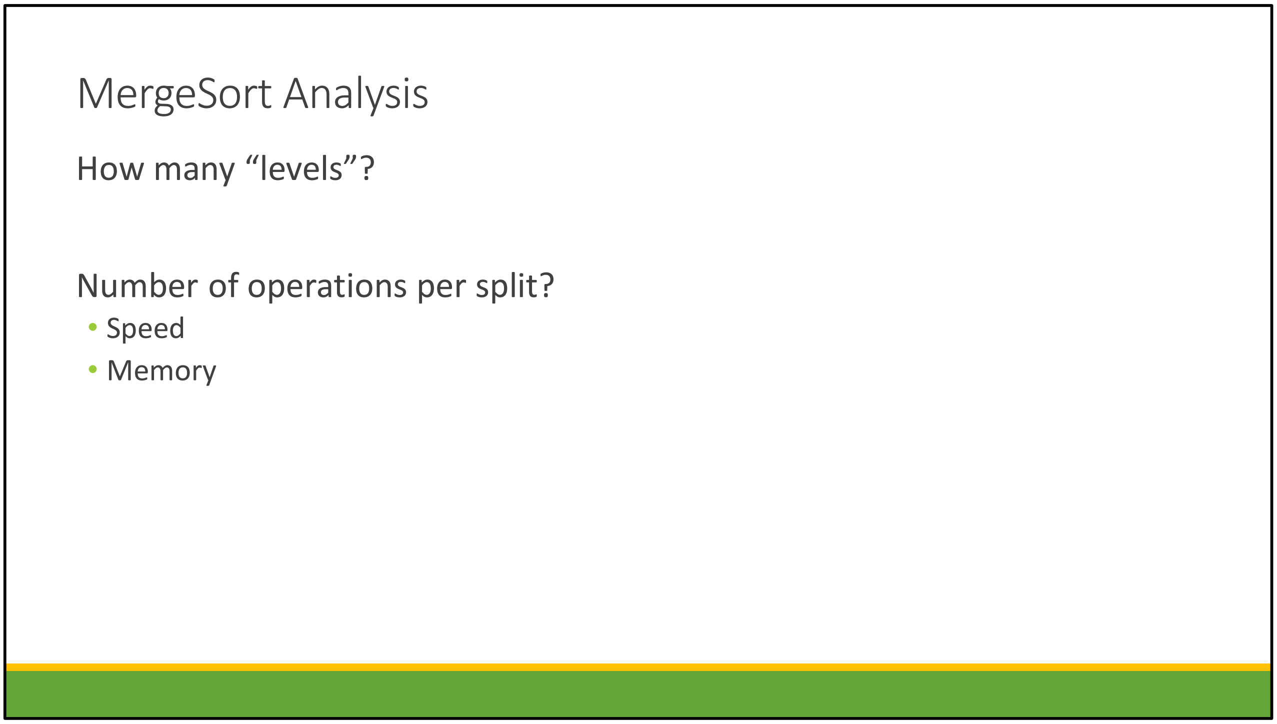
How to MergeSort an array?

- Split data in half, sort each half, merge results
- How to sort each half? THINK RECURSIVELY!

57	50	31	84	68	84	76	-4	48	33	84	-2	82	72	14	1

Need extra memory to store merged results

Look at code, in detail. RUN!



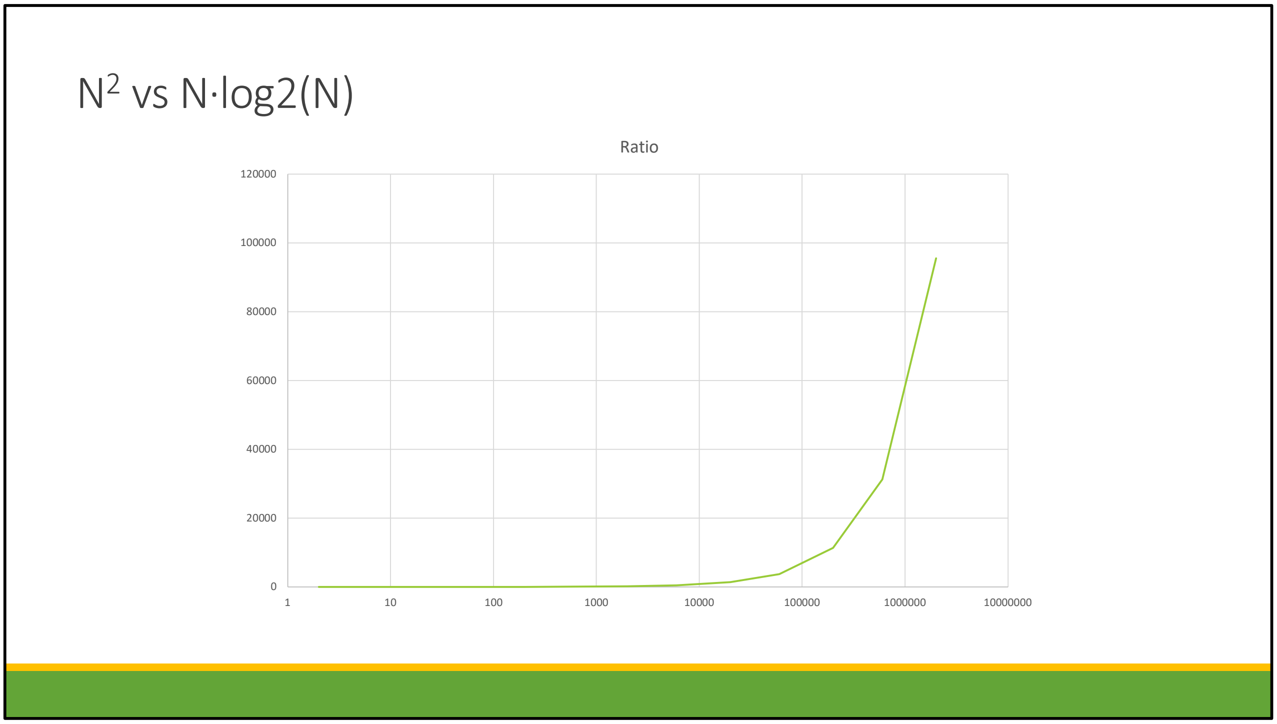
MergeSort Analysis

How many “levels”?

Number of operations per split?

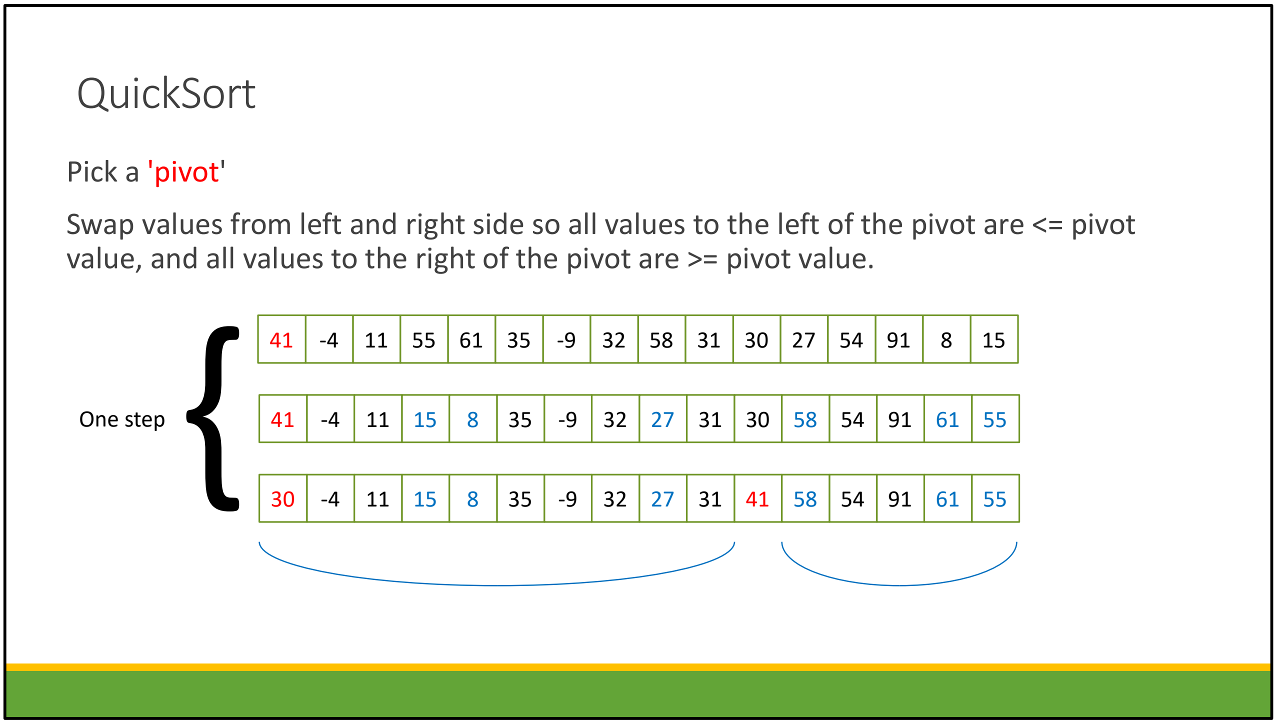
- Speed
- Memory

Assume N = 2^k. Layers are N, N/2, N/4, … 0. log2(N) layers.
3N comparisons per layer, 2N data copies. Total cost = O(N log2(N))
Uses N+N/2+N/4+N/8+… = 2N extra memory



N^2 vs N·log2(N)

For 1M elements, BubbleSort takes 60k x longer than MergeSort



QuickSort

Pick a 'pivot'

Swap values from left and right side so all values to the left of the pivot are <= pivot value, and all values to the right of the pivot are >= pivot value.

One step

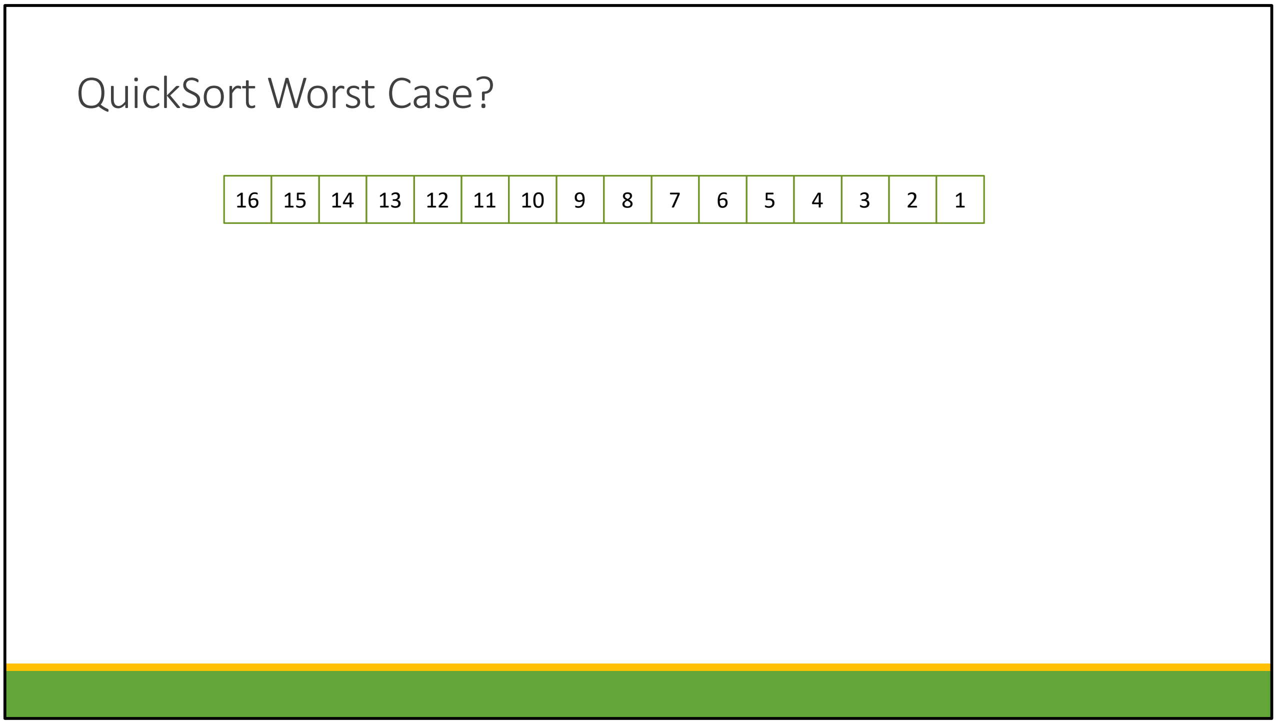
41	-4	11	55	61	35	-9	32	58	31	30	27	54	91	8	15
41	-4	11	15	8	35	-9	32	27	31	30	58	54	91	61	55
30	-4	11	15	8	35	-9	32	27	31	41	58	54	91	61	55

After swapping, PIVOT ELEMENT IN PROPER PLACE!

HOW MANY OPS?

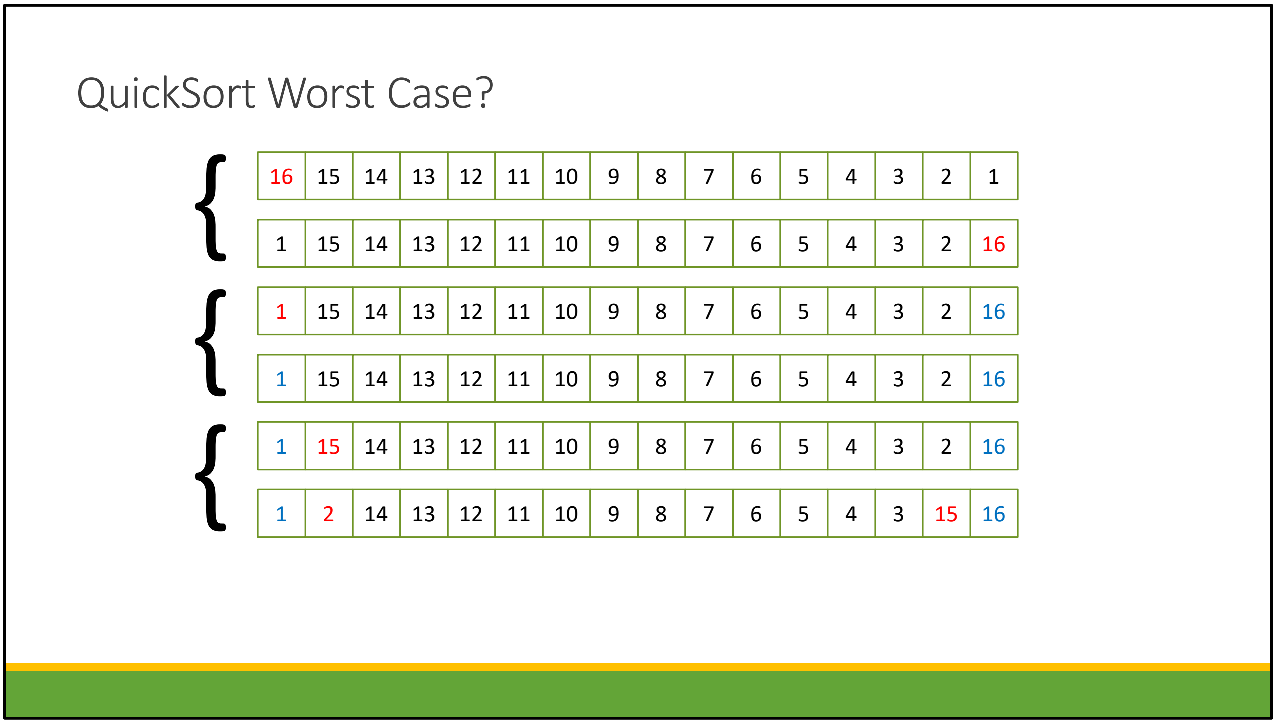
- At each step, N compars & swaps
- LOG2(N) steps like MergeSort. So O(N log2(N))

WALK THRU SW & RUN



QuickSort Worst Case?

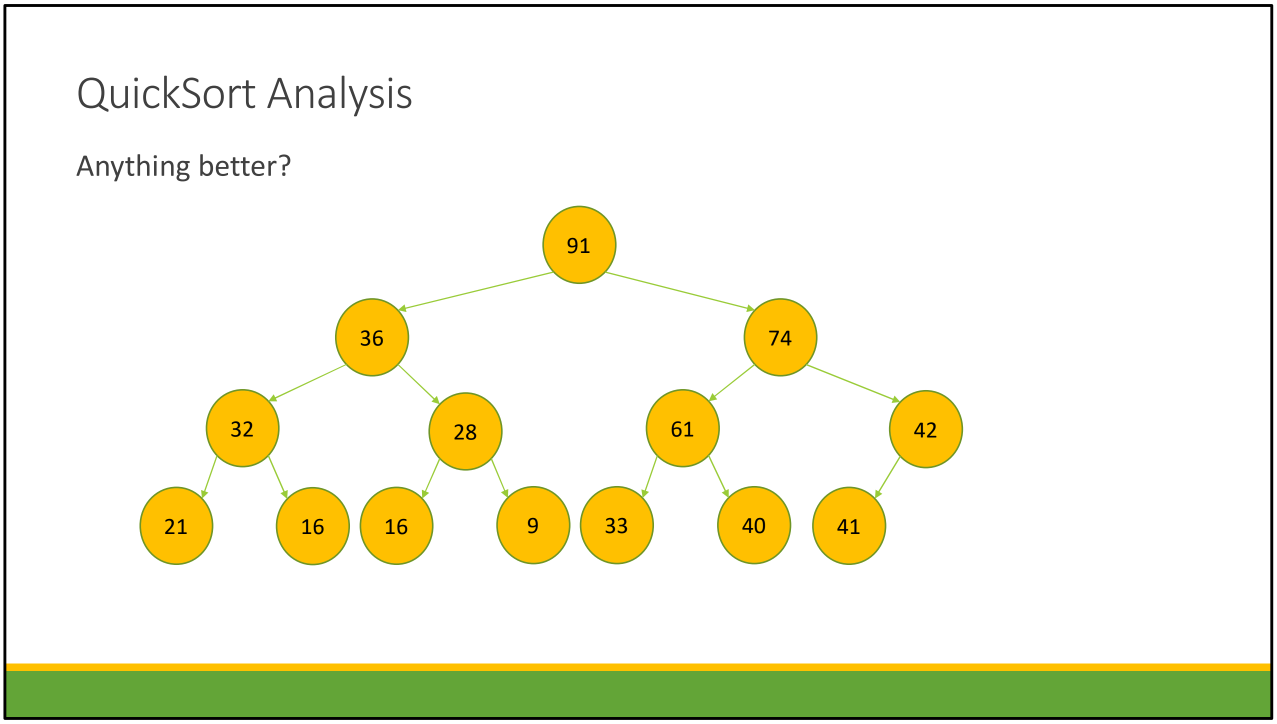
16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1



QuickSort Worst Case?

16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1
1	15	14	13	12	11	10	9	8	7	6	5	4	3	2	16
1	15	14	13	12	11	10	9	8	7	6	5	4	3	2	16
1	15	14	13	12	11	10	9	8	7	6	5	4	3	2	16
1	15	14	13	12	11	10	9	8	7	6	5	4	3	2	16
1	2	14	13	12	11	10	9	8	7	6	5	4	3	15	16

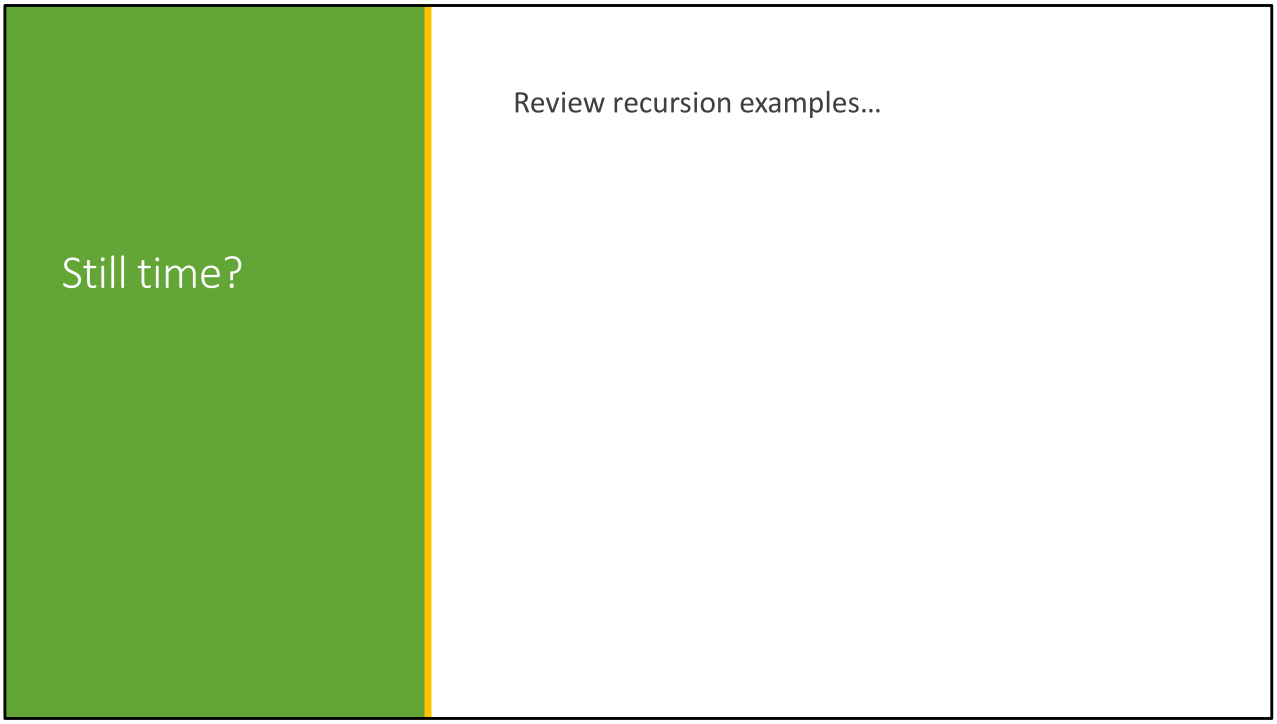
What are pivots? How many steps? QuickSort is O(N log2(N)) on average. Is O(N*N) in worst case, like SelectionSort.
RUN PROG WITH ORDERED INPUT! CRASHES!



QuickSort Analysis

Anything better?

HEAPSORT
Worst case O(Nlog(N))
No memory needed
But average time > QuickSort & MergeSort. Worst case very good though.



Still time?

Review recursion examples…

See sample problems in Piazza